PTO/SB/44 (10-96)
Approved for use through 6/30/99. OMB 0651-0033
Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB Control number
(Also Form PT-1050)

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

Page 1 of 1

PATENT NO.   : 7,398,553

APPLICATION NO. : 10/042,804

ISSUE DATE   : July 8, 2008

INVENTOR(S)   : Jianghao LI

It is certified that an error appears or errors appear in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

In the Claims:

Col. 12, line 5, (Claim 2)   change "sctrce" to --source--.

Col. 12, line 41, (Claim 9) change "tread" to --thread--.

Col. 13, line 3, (Claim 15) after "claim 13," insert --further--.

Col. 13, line 27, (Claim 19) change "head" to --thread--.

Col. 14, line 16, (Claim 23) change "had" to --thread--.

MAILING ADDRESS OF SENDER (Please do not use customer number below):

Jonathan O. Scott
BEYER LAW GROUP LLP
P.O. Box 1687
Cupertino, CA 95015-1687

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Jianghao LI	Attorney Docket No.: TRNDP005/TM00-003US
Patent: 7,398,553	Issued: July 8, 2008
Title: SCRIPTING VIRUS SCAN ENGINE	Confirmation No. 5044

REQUEST FOR CERTIFICATE OF CORRECTION OF OFFICE MISTAKE
(35 U.S.C. §254, 37 CFR §1.322)

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450
Attn: **Certificate of Correction**

Dear Sir:

Attached is Form PTO-1050 (Certificate of Correction) which is suitable for printing.

The errors together with the exact page and line number where the errors are shown correctly in the application file are as follows:

In the Claims:

Col. 12, line 5, (Claim 2)	change "sctrce" to --source--. This appears correctly on page 2 of the amendment filed May 26, 2006 in claim 2, line 2.
Col. 12, line 41, (Claim 9)	change "tread" to --thread--. This appears correctly on page 4 of the amendment filed May 26, 2006 in claim 11, line 8.
Col. 13, line 3, (Claim 15)	after "claim 13," insert --further--. This appears correctly on page 5 in the amendment filed May 26, 2006 in claim 17, line 1.
Col. 13, line 27, (Claim 19)	change "head" to --thread--. This appears correctly on page 6 of the amendment filed May 26, 2006 in claim 21, last line.
Col. 14, line 16, (Claim 23)	change "had" to --thread--. This appears correctly on page 7 of the amendment filed May 26, 2006 in claim 25, line 11.

Patentee hereby requests expedited issuance of the Certificate of Correction because the errors lie with the Office and because the errors are clearly disclosed in the records of the Office. As required for expedited issuance, enclosed is documentation that unequivocally supports the patentee's assertion without needing reference to the patent file wrapper.

It is noted that the above-identified errors were printing errors that apparently occurred during the printing process. Accordingly, it is believed that no fees are due in connection with the filing of this Request for Certificate of Correction. However, if it is determined that any fees are due, the Commissioner is hereby authorized to charge such fees to Deposit Account 50-4481 (Order No. TRNDP005).

Respectfully submitted,
BEYER LAW GROUP LLP

/Jonathan O. Scott/
Jonathan O. Scott
Registration No. 39,364

P.O. Box 1687
Cupertino, CA 95015-1687
408-255-8001

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: Li, Jianghao	Attorney Docket No.: TRNDP005
Application No.: 10/042,804	Examiner: ZIA, Syed
Filed: October 29, 2001	Group: 2131
Title: SCRIPTING VIRUS SCAN ENGINE	Confirmation No.: 5044

REPLY C

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

In response to the Office Action dated February 28, 2006, a response to which is due May 28, 2006, please enter the following amendments and remarks:

Amendments to the Claims are reflected in the listing of claims which begins on page 2 of this paper.

Remarks begin on page 8 of this paper.

1.   (previously presented)   A method for identifying a computer virus in interpreted language source code using a virus scan engine, the method comprising:

     receiving a portion of interpreted language source code;

     generating a language-independent representation of the portion of the interpreted language source code;

     comparing the language-independent representation with a virus signature in a pattern matcher; and

     determining if the language-independent representation matches the virus signature, whereby a match indicates a computer virus has been identified.

2.   (Original)   The method of claim 1, wherein the interpreted language source code is a scripting language source code.

3.   (Original)   The method of claim 1, wherein the virus signature is a language-independent representation of an interpreted language source code computer virus.

4.   (Original)   The method of claim 1, wherein the portion of interpreted language source code and the virus signature are represented as a linearized string of key actions.

5. (Currently amended) A method for generating a virus signature using a virus scan engine, the method comprising:

receiving a portion of interpreted language source code containing a computer virus;

parsing the portion of interpreted language source code into tokens using a parser;

inputting at least a portion of the tokens into a threadizor;

generating a language-independent representation of the computer virus from said portion of the tokens using said [[a]] threadizor, wherein said language independent representation is a linearized string of key actions; and

storing the language-independent representation of the computer virus as a virus signature.

generating the language-independent representation of the computer virus using

6. (Original) The method of claim 5, wherein the interpreted language source code is a scripting language source code.

7. (Original) The method of claim 5, wherein the virus signature is compiled in binary format.

8. (cancelled)

9. (Original) The method of claim 5, wherein the virus signature includes input from a virus analyst.

10. (cancelled)

11.   (previously presented)   A method for identifying a virus in interpreted language source code using a virus scan engine, the method comprising:

receiving a portion of interpreted language source code;

parsing the portion of the interpreted language source code into tokens to generate a tokenized source code using a parser, wherein at least some of the tokens represent key actions;

extracting selected key actions from the tokenized source code,

linearizing the key actions to generate an executing thread;

comparing the executing thread with a virus signature of a known virus in a pattern matcher; and

determining whether the executing thread matches the virus signature.

12.   (Original)   The method of claim 11, further comprising:

outputting the identification of the known virus.

13.   (Original)   The method of claim 11, wherein the portion of the interpreted language source code is lexically parsed.

14.   (Original)   The method of claim 11, wherein the portion of the interpreted language source code is lexically and grammatically parsed.

15. (previously presented)   A method using a virus scan engine for generating a virus signature from a portion of interpreted language source code including a computer virus, the method comprising:

receiving a portion of interpreted language source code containing a computer virus;

parsing the portion of the interpreted language source code containing the computer virus into tokens to generate tokenized source code using a parser, wherein at least some of the tokens represent key actions;

extracting key actions from the tokenized source code,

linearizing the key actions to generate an executing thread using a threadizor;

determining the set of minimum key actions in the executing thread required to effect the computer virus; and

storing the set of minimum key actions as a virus signature.

16. (Original)   The method of claim 15, further comprising:

compiling the virus signature in binary format.

17. (Original)   The method of claim 15, further comprising:

compiling the virus signature with data input by a virus analyst; and

storing the virus signature as part of a virus pattern file.

18. (Original)   The method of claim 17, wherein the virus pattern file further includes a dictionary of key actions.

19. (Original)   The method of claim 15, wherein the portion of the interpreted language source code is lexically parsed.

20. (Original)   The method of claim 15, wherein the portion of the interpreted language source code is lexically and grammatically parsed.

21.   (previously presented)   A computer readable medium containing program code for identifying a computer virus in interpreted language source code using a virus scan engine, the computer readable medium comprising instructions for:

receiving a portion of interpreted language source code;

parsing the portion of the interpreted language source code into tokens to generate a tokenized source code using a parser, wherein at least some of the tokens represent key actions;

linearizing at least a portion of the key actions to generate an executing thread;

comparing the executing thread with a virus signature of a known computer virus in a pattern matcher; and

determining whether the executing thread matches the virus signature.

22.   (Original)   The computer readable medium of claim 21, further comprising:

outputting the identification of the known computer virus.

23.   (Original)   The computer readable medium of claim 21, wherein the portion of the interpreted language source code is lexically parsed.

24.   (Original)   The computer readable medium of claim 21, wherein the portion of the interpreted language source code is lexically and grammatically parsed.

25.   (previously presented)   A computer readable medium containing program code using a virus scan engine for generating a virus signature from a portion of interpreted language source code including a computer virus, the computer readable medium comprising instructions for:

   receiving a portion of interpreted language source code containing a computer virus;

   parsing the portion of the interpreted language source code containing the computer virus into tokens to generate tokenized source code using a parser, wherein at least some of the tokens represent key actions;

   linearizing at least a portion of the key actions to generate an executing thread using a threadizor;

   determining the set of minimum key actions in the executing thread required to effect the computer virus; and

   storing the set of minimum key actions as a virus signature.

26.   (Original)   The computer readable medium of claim 25, further comprising:
   compiling the virus signature in binary format.

27.   (Original)   The computer readable medium of claim 25, further comprising:
   compiling the virus signature with data input by a virus analyst; and
   storing the virus signature as part of a virus pattern file.

28.   (Original)   The computer readable medium of claim 27, wherein the virus pattern file further includes a dictionary of key actions.

29.   (Original)   The computer readable medium of claim 25, wherein the portion of the interpreted language source code is lexically parsed.

30.   (Original)   The computer readable medium of claim 25, wherein the portion of the interpreted language source code is lexically and grammatically parsed.

REMARKS

Claims 1-7, 9, and 11-30 are still pending in this application. Claims 8 and 10 have been incorporated into independent claim 5. Claims 8 and 10 have been cancelled. Reconsideration of the application is earnestly requested.

<u>Response to Office Action</u>

The Examiner has maintained the rejection of the claims in view of *Chess* and *Chandnani*. In the recent final office action (page 4) and in the current office action, it is stated that Applicants have failed to explicitly identify specific claim limitations which would define a patentable distinction over prior art. To the contrary, <u>Applicant submits that the Reply mailed June 9, 2005 very specifically identified at least one claim limitation in each of the independent claims that was not present in any of the cited art</u>.

Applicant detailed very carefully why the *Chess* and the *Chandnani* references are not relevant and pointed out which elements of the claims the references do not teach or suggest. For example, the three steps of claim 5 not present in *Chess*. Claim 15 requires extracting, linearizing and determining key actions; *Chess* does not teach or suggest such steps. Claim 1 requires "generating a language independent representation of the portion of the interpreted language source code." *Chandnani* does not teach or suggest generating a language independent representation of interpreted language source code. Claim 11 requires extracting and linearizing key actions and comparing the executing thread; the steps are not taught or suggested by *Chandnani*. Claims 21 and 25 are Beauregard claims that are similar to the above claims. Various of the dependent claims also contain features not present in the cited art as detailed in the previous Reply.

<u>Response to Arguments--*Chess*</u>

In the current Office Action the Examiner presents a Response to Arguments starting at page 3 in which specific sections of the prior art are relied upon as disclosing features of the independent claims.

The current Office Action now cites column 4, line 38 to column 5, line 2 and column 11, lines 39 to 56 of *Chess* as disclosing claims 5, 7-10, 15-20 and 25-30. But, columns 4 and 5 only disclose a comparison of a host computer file and an infected version of the host computer file.

A virus-attachment description and a virus-structure description are prepared, but the required features of the independent claims are not disclosed.

Claim 5

Claim 5 requires "<u>receiving a portion of interpreted language source code containing a computer virus</u>." *Chess* does not teach or suggest an interpreted language that contains a computer virus nor source code that contains a virus. That portion of *Chess* relied upon only discloses host and infected files written in binary code or bytes. There is no discussion of source code or of an interpreted language.

Claim 5 also requires "<u>parsing the portion of interpreted language source code into tokens using a parser</u>." The computer virus code shown in Figure 4 of *Chess* is not interpreted language source code that is parsed into tokens.

Claim 5 also requires "<u>generating a language-independent representation of the computer virus from said portion of the tokens</u>." That portion of *Chess* relied upon does not disclose generating a representation of the computer virus that is independent of the language in which is written. In fact, Figure 4 simply shows manipulation of byte codes, the language in which viruses Sample 1 and Sample 2 are written. A byte code is not independent of the language used.

Claim 5 also requires that "<u>the language independent representation is a linearized string of key actions</u>." The Office Action relies upon column 12 of *Chess* for this feature. But this portion of *Chess* is referring to Figure 4 that simply shows that a computer virus is separated into sections of byte codes, some being variable over samples, some being constant. A byte code is not a "key action" as described in the Specification of the instant application; nor is the string "linearized." As described in the Specification of the instant application, "linearized" refers to rearranging portions of the virus code such that they appear in the order in which they are executed; Figure 4 simply shows that the byte codes of the computer virus are left as is.

Claim 5 also requires "<u>storing the language-independent representation of the computer virus as a virus signature</u>." Because *Chess* does not disclose a language-independent representation, it likewise does not disclose storing that representation as a virus signature. In fact, *Chess* does not concern itself with generating in storing a virus signature for future use. Column 11, lines 55-57 mentions that a signature can be extracted for identification, but there is no teaching that this signature is stored for later use.

Claim 15

Those features of claims 15-20 similar to the above features of claims 5-10 that have already been discussed are likewise not taught or suggested by *Chess* for the above reasons.

Further, claim 15 requires "<u>extracting key actions</u>," "<u>linearizing key actions</u>," "<u>determining the set of minimum key actions</u>," and "<u>storing the set of minimum key actions as a virus signature</u>." Respectfully, it is pointed out that columns 11 and 12 of *Chess* do not involve extracting, linearizing or determining key actions. Again, it is pointed out that Figure 4 of *Chess* simply discloses placing the virus code into sections and determining which sections remain constant over samples and which sections are variable. None of the above-cited steps of claim 15 are taught or suggested in *Chess*.

Response to Arguments--*Chandnani*

The current Office Action now cites paragraphs 16, 20, 29, 51-56 and 64 of *Chandnani* as disclosing claims 1-4, 11-14 and 21-24. But, paragraph 16 only discloses detecting a script language virus using language description data <u>dependent</u> on the script language. Claim 1 requires a language <u>independent</u> representation. Paragraph 20 only discloses that the incoming data stream (the script language) is converted into a stream of tokens; paragraph 29 discloses lexical analysis of a data stream. Claim 11, though, requires further steps as explained below. Paragraphs 51-56 (and paragraph 50) describe creating viral code detection data (a detection regimen) from samples of script language viral code. The detection regimen is converted to binary format and then matched against suspected viral code (such as a corrupt system macro, as shown). But, the independent claims require distinctive features as explained below.

Claims 1-4

Claim 1 is a method for identifying a computer virus in interpreted language source code. After a portion of the code is received, the second step of the claim requires "<u>generating a language independent-representation of the portion of the interpreted language source code</u>." The third and fourth steps also require the "language-independent representation" limitation. Use of a language-independent representation is beneficial because it can better detect polymorphs of scripting language viruses.

By contrast, *Chandnani* does not teach or suggest generating a language-independent representation of the interpreted language source code to be scanned for a virus. *Chandnani* does show in Figure 2 that an incoming data stream representing script code is fed into a detection engine 53 that has a lexical analyzer. (Paragraphs 57-62.) The lexical analyzer converts the data stream into a stream of tokens but no other processing is performed on the tokens. A stream of tokens is not a language independent representation of the source code. A stream of tokens is very much a language dependent stream of characters, language constructs, symbols, etc., that represent a particular scripting language.

Because the second step of claim 1 is not taught or suggested, it is requested that the rejection be withdrawn.

Claim 3 requires that "the virus signature is a language independent representation of an interpreted language source code computer virus." The data stream of tokens in *Chandnani* is compared to the virus detection data from database 57. But the virus detection data is not a language independent representation of source code. The virus detection data is a pattern of tokens or a CRC signature check (Paragraph 51), neither of which is a language independent representation. Further, because the virus detection data is compared directly to the stream of tokens (the tokens representing the particular script language used), the virus detection data must also be in the form of the particular script language being used (Paragraph 64). If the virus detection data were in a *language independent* form it would not be possible to match the stream of tokens which are in a *language dependent* form. The Office Action cites paragraph 55 as disclosing that the virus signature is in a language independent representation, but paragraph 55 only discloses that DFA data is used to transition from one stage in a pattern match operation to the next. The actual pattern match data (i.e., the virus signature) is still in a language dependent form.

Claim 4 requires that both the source code and the virus signature "are presented as a linearized string of key actions." As discussed above, *Chandnani* does not disclose key actions nor a linearized string.

Claims 11-14

Claim 11 requires "extracting selected key actions from the tokenized source code," "linearizing the key actions to generate and executing thread," and "comparing the executing thread with a virus signature of a known virus." As pointed out above in the overview of

Paragraphs 57-64 only disclose that the incoming data stream is converted into a stream of tokens by a lexical analyzer; no further conversion or processing is performed. Thus, the required steps of "extracting selected key actions," "linearizing the key actions," and "comparing the executing thread" are not taught or suggested by *Chandnani*. The Office Action relies upon paragraph 20, but paragraph 20 only discloses that the incoming data stream (the script language) is converted into a stream of tokens. The Office Action also cites paragraph 64 but this paragraph only discloses that a pattern is matched against the token stream; there is no "executing thread" because *Chandnani* does not linearize any key actions let alone the generate key actions from the tokens in the first place.

Therefore, Applicant submits that the claims are allowable and requests that the rejections be withdrawn.

Reconsideration of this application and issuance of a Notice of Allowance at an early date are respectfully requested. If the Examiner believes a telephone conference would in any way expedite prosecution, please do not hesitate to telephone the undersigned at (612) 252-3330.

Respectfully submitted,
BEYER WEAVER & THOMAS

Jonathan O. Scott
Registration No. 39,364

BEYER WEAVER & THOMAS, LLP
P.O. Box 70250
Oakland, CA 94612-0250

Telephone: (612) 252-3330
Facsimile: (612) 825-6304